Touché: Data-Driven Interactive Sword Fighting in Virtual Reality

Javier Dehesa[1] **Andrew Vidler**[2] **Christof Lutteroth**[1] **Julian Padget**[1]
[1]University of Bath, UK [2]Ninja Theory Ltd., UK
{J.Dehesa, C.Lutteroth, J.A.Padget}@bath.ac.uk

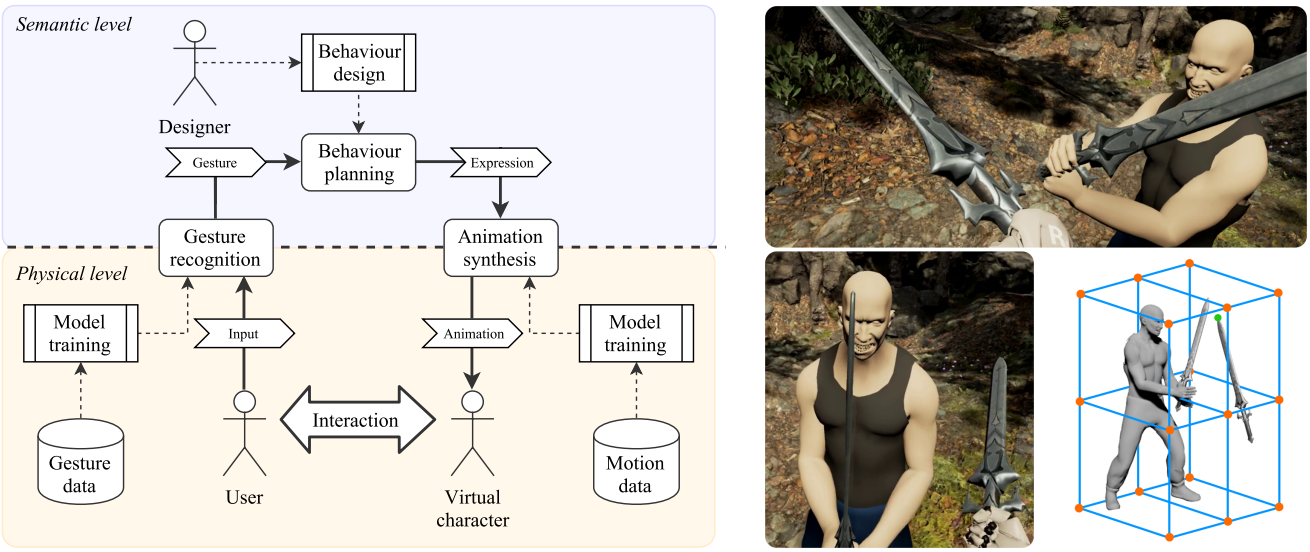

Figure 1: Left: Our framework splits the problem of simulating interactive VR sword fighting characters into a "physical" level, relying on data-driven models, and a "semantic" level, where designers can configure the behaviour of the character. Right: The framework generates responsive animations against player attacks (top), avoiding nonreactive behaviour from the character (bottom left). A neural network parameterised by the position of the player's sword synthesises the animation (bottom right).

ABSTRACT

VR games offer new freedom for players to interact naturally using motion. This makes it harder to design games that react to player motions convincingly. We present a framework for VR sword fighting experiences against a virtual character that simplifies the necessary technical work to achieve a convincing simulation. The framework facilitates VR design by abstracting from difficult details on the lower "physical" level of interaction, using data-driven models to automate both the identification of user actions and the synthesis of character animations. Designers are able to specify the character's behaviour on a higher "semantic" level using parameterised building blocks, which allow for control over the experience while minimising manual development work. We conducted a technical evaluation, a questionnaire study and an interactive user study. Our results suggest that the framework produces more realistic and engaging interactions than simple hand-crafted interaction logic, while supporting a controllable and understandable behaviour design.

CHI '20, April 25–30, 2020, Honolulu, HI, USA

ISBN 978-1-4503-6708-0/20/04…$15.00
DOI: 10.1145/3313831.3376714

Author Keywords

virtual reality; sword fighting; machine learning; animation; gesture recognition

CCS Concepts

•**Human-centered computing** → **Virtual reality;** •**Computing methodologies** → *Machine learning; Animation;*

INTRODUCTION

As VR takes its place as a commoditised medium accessible to end users in their own homes, expectations about the possibilities of the technology are raised. From a user perspective, VR should allow us to transform traditional screen-based productions into embodied experiences, where one can directly interact with the virtual world from "inside" [46]. However, designing the mechanics for the virtual world is not always straightforward. Close-range interactions are particularly difficult, due to the unpredictability of user actions, and conventional methods used in non-immersive environments, like screen-based video games, extrapolate poorly to VR.

In this work, we study the case of sword fighting scenarios. Sword fighting has been always present in some way throughout video games history [53], and it is a representative example of an engaging, embodied experience for bringing into VR. It is also a good example of the intrinsic challenges that VR entails. In a modern screen-based video game, an interactive sword fighter is usually, broadly speaking, driven by scripted logic and animation clips (e.g. pressing an "attack" button triggers a sequence of animations). In VR, however, the problem becomes more complex to model. On the one hand, player input is more difficult to process. Unlike in the case of a conventional controller, control in VR is enabled through tracking hardware. If a player wants to strike the opponent with a sword, they just need to move their hand accordingly. This is much more intuitive and immersive for the player, but it introduces the problem of deciding whether a strike is taking place or not. If a designer wants to trigger some specific reactions after a particular strike, it is important to be able to tell whether the strike actually happened or it was just a slight hand movement. Equally problematic is that the space of possible situations that the system may need to handle becomes virtually unlimited: since the actions of the player are not constrained to predefined patterns, while the reactions from the opponent must adapt to the arbitrary sword trajectories from the player. In this sense, accounting for every possible situation in a hand-made system becomes infeasible.

We propose data-driven methods as a scalable and effective alternative for developing interactive VR systems. In particular, we investigate whether (a) it is possible to build an interactive VR sword fighting scenario using data-driven techniques, (b) it can be done in a resource-effective manner, and (c) it can be done to an acceptable level of quality. We thus present Touché, a data-driven framework to model sword fighting interaction in VR [13]. Our framework is capable of recreating the actions of a sword fighter wielding a two-handed sword, alternating between defensive and offensive actions, while reacting to the actions of the player. We draw on ideas in gesture recognition and animation synthesis to build a complete interaction system that abstracts away low-level complexity, yet allows for effective control of the overall behaviour. This reduces the design work to configuring a set of parameters in the framework. Specifically, our contributions are:

- A data-driven framework for real-time VR sword fighting that automates the most complex design aspects and minimises the amount of necessary human work.
- A methodology to collect data, train and integrate the data-driven models required by said framework.
- A thorough evaluation of our proposal, both from technical and user-oriented perspectives.

RELATED WORK

This section reviews relevant literature across several fields related to our research, namely virtual humans, gesture recognition, artificial reasoning and automatic character animation.

Virtual Humans

Building virtual human-like characters capable of interacting with real users in a natural fashion has been a long-standing research topic. Several studies suggest that virtual humans may produce some similar reactions as real ones in users [2, 43, 35], and immersive interaction with virtual characters has been shown to be useful for therapeutic [31], educative [28] and artistic applications [3]. To a large extent, the design of virtual humans has been focused on verbal skills. This is a natural consequence of typical goals of the agents, such as education [58] or communication [52]. More sophisticated virtual humans try to go further by extracting more information from the user beyond words. This is the case of SimSensei [14] a feature-rich virtual interlocutor designed to assess distress indicators related to different psychological disorders. It includes multiple nonverbal communication perception mechanisms (facial expression, gaze direction, etc.), speech recognition and empathetic body language. However, the agent was never designed for an immersive environment or complex interactions at close distance. One of the most remarkable examples of virtual humans is the BabyX project [60], a virtual baby capable of reacting to visual input from humans with facial animation and imitation learning, among other things. The project is focused on biologically plausible animation and mind modelling, developing complex human-like nonverbal behaviours. As of now, though, they have not investigated direct interaction with objects and humans in a virtual world.

Overall, work in this area is focused on the cognitive aspects of the character, but they do not consider complex interactions as much, and in particular they are generally not designed for interactive VR. They are also largely hand-crafted and do not contemplate the use of data-driven methods to aid the design.

Gesture Recognition

We intend to use gesture recognition as a means to interpret the actions of the user. This topic encompasses a varied set of problems that have been a subject of study for a long time [47]. The most common model in gesture recognition is the hidden Markov model (HMM), as proposed first by Yamato et al. [74]. This approach is based on feature vectors extracted from the input, so it is a flexible framework that can potentially work with any kind of data. However, HMMs are not immediately applicable in real time, firstly because they require an analysis of the whole gesture sequence before yielding a classification, but more importantly because, given a stream of data containing multiple gestures, they are not able to segment it into individual gestures. Explicitly demarcating the boundaries of each gesture [30, 45, 61, 15] is not an acceptable solution in our context, and existing heuristics to detect gesture boundaries [42, 33, 55] are not easily extrapolated to two-handed gestures. Yin and Davis [75] extend the capabilities of HMMs with a hierarchical model that supports continuous gesture recognition with a small delay, even providing basic information about the progression of the gesture. Other extensions of the HMM model include hidden conditional random fields (HCRF) [68] or hidden conditional neural field (HCNF) [44], which do extend the capabilities of the method but not in terms of real-time behaviour. Another popular time-based model used in this context is dynamic time warping (DTW) [32, 57]; however, DTW cannot in general be used over arbitrarily long streams with multiple gestures and, like other approaches, can only recognise already completed gestures. The interested

reader can refer to the survey by Cheng et al. [9] for a deeper review of these models. More general methods have also been applied to the problem, such as neural networks [69, 73], which have been particularly successful in video-based applications [51, 17, 66, 49, 1]. Support vector machines (SVM) have also been used [18], with their associated benefits of low memory and computational requirements. Nevertheless, the application of these techniques to sensor-based gesture recognition has been far less studied. Given the limited complexity of our features (two hand sensors and one head sensor), we chose neural networks to implement gesture recognition in our framework for their simplicity and proven effectiveness.

Artificial Reasoning

When it comes to modelling the behaviour of a virtual agent, there is a range of possibilities that can be explored. There is a long history of computer emulation of complex human reasoning as a model of agent behaviour. Classic early examples are planning strategies like the STRIPS algorithm [19], later extended into hierarchical task-network (HTN) planning [16], which allows agents to come up with plans of actions towards specific goals in mostly static and predictable environments. For real-time, dynamic contexts, arguably the most commonly referred kind of agent architecture is belief–desire–intention (BDI), proposed first by Bratman [4] and then Rao and Georgeff [56]. This framework describes agents as a compound of some knowledge about the world, or beliefs, one or more goals, or desires, and a set of plans. At any instant, the agent updates its belief base and considers which plan better satisfies its desires, so it can always accommodate for dynamic changes and plan failures. Similar models include Procedural Reasoning System (PRS) [21], a particular implementation of BDI with multiple real-world applications, and belief–obligation–desire–intention (BOID) framework [5], which introduces the concept of obligations into the paradigm. More comprehensive proposals can be found under the class of cognition architectures, which have been under research for a number of decades as well [36]. These attempt to provide principled models that reproduce either the entirety or a significant part of human intelligence (learning, reasoning, etc.). Even though some of these models, such as SOAR [39], have seen durable success in some contexts, they are generally far too complicated for agents attempting to solve well-defined, specific tasks only.

For our purposes, we implement a basic behavioural planner based on a simple state machine in our framework, although our approach could support more sophisticated reasoning models to model scenarios that require them.

Character Animation Synthesis

Real-time character animation remains a notorious challenge in high-quality digital productions. Mature motion capture technology enables absolutely realistic animation, but working with it in interactive real-time media is not without challenges. Effective manipulation techniques for motion capture data have been studied for a long time now [6, 70, 22], but producing a continuous stream of fluid motion from a collection of motion clips in real time involves a new level of complexity, since it requires the generation of natural transitions between independent animation sequences in real time. Perlin [54] outlined an early proposal for an animation framework based on mixtures or blends of multiple hand-crafted actions. Today, the most common methodology is to manually design complicated state machines and "blend trees" expressing interpolation of multiple clips according to different combinations of user input and environmental information [48]. While this method allows for great control and predictability, it is very time-consuming and difficult to scale, and the results have limited re-usability. Kovar et al. [38] introduce the influential concept of motion graph as a means to automate locomotion animation. It is a graph storing short animation clips at its edges such that any path within it can be mapped into a smooth animation. This model has been used as a base for different extensions [23, 37] and as inspiration for entirely new designs. Treuille at al. [65] propose a comparable approach for bipedal motion using control theory, defining a structure akin to a motion graph and using optimal planning approximations. These works shaped the design of the crowd animation system in the video game *Hitman: Absolution* by IO Interactive [7]. A further development of the idea was the introduction of motion fields by Lee at al. [41], which can be roughly described as a continuous version of a motion graph, with animations embedded in a high dimensional vector space. Ubisoft Montreal developed a simplified version of motion fields, called motion matching, for their video game *For Honor* [11].

These are, however, non-parametric models, meaning that their complexity increases with the size of the data. Neural networks have been found to be a plausible parametric alternative, offering favourable traits in terms of memory and computational cost. Taylor et al. [64] and Fragkiadaki et al. [20] show different neural models able to apprehend the patterns in a motion capture database and generate streams of animation in real time. Recently, Holden et al. [27] designed an interactive animation system using a phase-functioned neural network architecture, where each weight of the network is replaced by a closed spline function evaluated at a "phase" point. The phase represents the location of the current pose in a walking cycle, so the proposal is specifically oriented to locomotion. The system is able to reproduce highly realistic animations on a variety of environments with multiple styles. Similarly, Starke et al. [62] propose a system of specialised neural networks to model the locomotion of a quadruped. These works focus on character control and realistic reactions to a static environment, but do not handle interaction with another dynamic avatar, and in particular VR interaction.

A distinct variant of the problem of character animation is that including interactions between a human and a character in proximity. Lee and Lee [40] used unlabelled motion capture data to produce control policies for characters in dynamic environments. Using a reinforcement learning approach, the authors are able to emulate boxing or tennis playing scenarios, but not for real-time interaction. Ho and Komura [25] proposed a topological characterisation of wrestling animation for two virtual agents, allowing a user to interactively decide the actions of one of the characters. However, the model is not necessarily applicable to other interactions, nor it is suited to human motion input. Ho at al. [24] outline a more general

method for human–character interaction animation. It queries a motion database with the current state of the user and character to search for similar recorded clips, editing the retrieved motion to match it to the actual situation. The method is demonstrated with close dance interactions in VR. Using a statistical approach, Taubert et al. [63] construct a hierarchical model for real-time interactions between humans and virtual characters. Their approach is showcased in a character able to "high-five" a user in a virtual environment with different styles. Vogt at al. [67] present a data-driven method conceptually similar to Ho et al. [24], but with several improvements, including a better low-dimensional representation of the current queried pose and both spatial and temporal matching to stored data.

For this work, we build on the phase-functioned neural network model [27] to implement our animation synthesis subsystem, extending it to suit our sword fighting scenario.

SYSTEM DESIGN

We have developed Touché, a framework for real-time sword fighting interaction in VR that abstracts the low-level complexity of the problem while leaving room for hand-made behaviour design. The framework models a simple sword fighting interaction between the player and a virtual character built on exchanges of two-handed, directional sword strikes. The character will approach the player and try to block their incoming strikes, while also initiating attacks of its own. Figure 1 shows a high-level view of Touché. We model input from the player on the left side of the model, and animation of the character on the right, with the goal of recreating an engaging interaction between the two.

We split the problem into two levels. The *physical level* represents actual events and information in the virtual world. For our purposes, this means essentially input from the player and character animation. It is difficult to reason directly with this data, as it is mostly 3D geometrical information with little structure. We therefore use data-driven models to project it onto a *semantic level*, where the information is seen as discrete labels for virtual world events. On the player's side, a gesture recognition system interprets raw 3D input as sword fighting gestures. It becomes now simple to define custom logic to decide how to react to the actions of the user. This is done by the human-designed behaviour planning module. This allows for explicit direction of the overall interaction, even though we are using data-driven models to hide the complexity, hence minimising human work without sacrificing control. The output of this module is composed simply of requests for a particular behaviour from the character, such as "attack" or "defend". It is the data-driven animation synthesis module, the one responsible for turning those requests into actual animation, that maps back to the physical level again. We describe each of these activities in more detail in the following subsections.

Gesture Recognition

The purpose of the gesture recognition module is to parse the actions of the user as meaningful gestures relevant to the sword fighting interaction. The gestures of interest to us are directional sword strikes. Specifically, we consider eight strike directions: up, up–right, right, down–right, down, down–left, left, up–left. We trained a neural network as the basis of our recogniser. To this end, we first collected a dataset of gestural actions. This was done in a separate VR environment, where the player was presented with a signal indicating a gesture to perform, which they then did while pressing a button on the hand controller. This eliminates the need for manual labelling later, as the beginning and end of each gesture are automatically recorded. We also needed to be able to predict the progress of a gesture through time, in order to be able to react to actions in a timely manner, so we compute a continuous progress indicator for every frame, with values ranging from zero to one in each gesture execution. In total, we collected over 11 minutes of data, recorded at 90 fps (over 62 700 frames). Furthermore, we augmented this dataset [72] with small random perturbations in height, scale, speed, trajectory and orientation, synthesising six times more data. This enhanced the performance of our recogniser without having to collect additional data manually, and allowed us to use data collected from a single subject without significant impact on the result, while greatly simplifying the process.

Each frame sample contains a limited set of features, namely the position and orientation of head and hands, provided by the VR tracking hardware. However, we do not have a frame of reference for the overall body position. This is a problem, because gestures are relative to the body, so it is necessary to fix some "point of view" for the 3D input. Here we simply opted to use the tracked position of the head as reference, applying exponential smoothing to the position and orientation. This gives us a reasonable estimation of the body pose if we assume users do not turn their heads for prolonged periods.

The gesture recognition network takes as input the tracked position and rotation of each hand, expressed as a 3D vector and a quaternion. The outputs include the detected gesture class, or "no gesture" (nine probability values, one for each gesture class), and the estimated gesture progress, as a value from zero to one. The network is trained as a sequence-to-sequence model, producing a continuous stream of predictions for the user input. We use a one-dimensional convolutional architecture with two hidden layers, each with a filter size of 4 and 100 channels. The first layer applies a dilation factor [76] of 4, meaning that, in total, each output is predicted using a window of 16 frames of input. We took 70% of the data for training, leaving the rest for validation and testing. The model was trained for 6 200 epochs with a dropout of 0.5, using batches of 175 sequences, each containing 2 000 frames.

With respect to the gesture progression, we are mainly interested in two kinds of events: the beginning and the ending of each gesture. The gesture recognition module triggers these events when a gesture is detected with a progress of less, and more, than 50%, respectively. This allows the behaviour planning module to make decisions depending on whether a gesture is starting or finishing.

Behaviour Planning

The behaviour planning module is in charge of determining the actions that the virtual character shall perform, given the identified actions of the player. Unlike the other two modules,

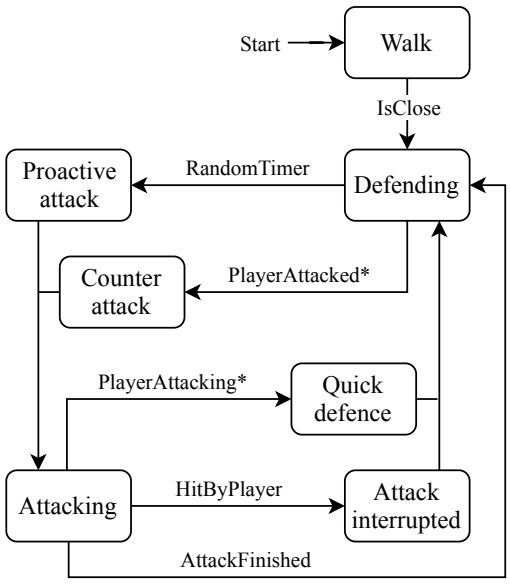

Figure 2: Character behaviour diagram. Transitions marked with * depend on the configuration of the framework.

which are based on data-driven processes, this one is human-designed, albeit at a very high level. The behaviour planning module implements the state diagram shown in fig. 2.

Initially, the character is at a distance from the player and starts walking in their direction until the distance is small enough (*IsClose*), from where it can either defend or attack. While defending, the character will simply try to put their sword across the trajectory of the player's sword. There are two circumstances that puts the character into attacking state: if no strikes from the user are detected for some random amount of time (*RandomTimer*), the character will proactively initiate an attack, or when a strike from the player is detected (*PlayerAttacked*), the character may react by counterattacking with a strike following the same direction, in an attempt to hit the unguarded area. For example, if the player performs a strike from right to left, then the right-hand side will be left unprotected, so the character may try to strike there.

When the character is attacking, it will usually continue the attack until it is completed (*AttackFinished*), and then come back to defend again, but the attack may also be aborted early. This will always happen when the player hits the character in the middle of an attack (*HitByPlayer*). But the character may also interrupt an attack willingly if an incoming strike from the player is detected (*PlayerAttacking*), quickly coming back to defending in an attempt to block the attack.

Though simple, this model exposes a few adjustable parameters that a designer may tune to direct the behaviour of the character. These define how exactly these transitions should take place. The first parameter is the average attack rate (s^{-1}), which regulates the aggressiveness of the character. This rate regulates the random time that the character waits between attacks (*RandomTimer*), sampled from an exponential distribution. The second parameter is the probability that the character reacts to an attack from the player with a counterattack (*PlayerAttacked*). This leads to a third parameter determining the average reaction time (s) between when a player's strike is completed and a counterattack is actually started. The actual delay is also sampled from an exponential distribution. Finally, a fourth parameter expresses the probability that an attack by the character is aborted due to an incoming attack from the player (*PlayerAttacking*).

Animation Synthesis

The animation synthesis module is responsible for determining the pose that the virtual character shall adopt in each frame. This module is driven by the directives emitted by the behaviour planning module, and can be seen as a translator of these directives into "physical" actions. Animation synthesis is also a data-driven component: it does not use complex state machines, but rather offers a menu of actions it may perform and acts them out, depending on the context. There are two kinds of actions that the model may perform: defending and attacking. Defending is the more complicated, as it is reactive and depends on what the player does. Therefore, the defence animation synthesis uses a machine learning model capable of generating the motion necessary to block arbitrary strikes from the user. Attacks, on the other hand, are initiated by the character by request of the behaviour planner, which also indicates the kind of attack to perform. For this, we can simply use a collection of animation clips that are played out as necessary. This simplifies the system and gives designers precise control of what happens when the character attacks the user. The animation synthesis system smoothly blends between the machine learning model and the clips, fading the weight of each one in and out over a short period of time as the character transitions between attack and defence, so the overall animation appears as a continuous action.

For the defence animation synthesis, we start by collecting a set of motion data to train the model. We used Vicon Bonita equipment to record several sword fighting attacks and blocks at different angles. We produced about 15 minutes of training data in total, recorded at 30 frames per second (over 26 000 frames), with each frame containing the pose of a 24-joint skeleton and the position of the tip of both swords. The data is split leaving 80% for training and the rest for evaluation.

Defence animation is generated one frame at a time. The model continuously predicts the next character pose using the current pose and the perceived control information from the user. In summary, the collection of features used as input is:

1. The latest position and orientation of the body joints, taken with respect to the root position of the character (between the feet). Orientations are given as two unit vectors in the direction of the rotated $\mathscr{X}$ and $\mathscr{Y}$ axes. This encoding is similar to the one used by Zhang et al. [77].
2. The current position and orientation of both swords. The position is taken from the tip, and in this case only one unit vector in the direction of the sword blade is used as orientation (the "roll" of the sword is ignored).
3. The recent trajectories of both swords. This is taken as the position and orientation of the sword three and six frames before the current frame. Having two recent snapshots

allows the model to capture patterns in the velocity and acceleration of the sword.
4. The closest points between the two swords. These are the points resulting from a 3D segment-to-segment distance geometrical query. The magnitude of the distance is also taken. This is useful to get the model to learn how to interpose the sword in the trajectory of an incoming attack.
5. The view angles of the opponent sword tip from the centre of gravity of the character. These are the yaw and pitch that orient the view in the direction of the sword from there.

Using these features, the system predicts the position and orientation of all the body joints and the sword from the character. At run time, each predicted pose is effectively fed back as input to the model for the next frame, defining an autonomous animation synthesis loop.

We chose to use a neural network as a model for this system, given their convenience and their success in comparable problems. We used an architecture inspired by the phase-functioned neural network by Holden et al. [27]. Instead, however, of training a single neural network, we train a collection, each specialised in particular aspects of the problem. We use a fixed network architecture, but train for multiple sets of weights (parameterisations). Each of these parameterisations is associated with a different location of the player's sword tip, meaning that they become active when the user's sword approaches the corresponding location. This idea is illustrated in the bottom right image in fig. 1. We define a 3D region where the interaction takes place between the virtual character and the player. The dimensions of this space can be adjusted depending on the data and the specific situation (character proportions, weapon size, etc.). We define an equally spaced grid in this 3D region, with every point in the grid having a corresponding parameterisation. For prediction, instead of simply picking the closest parameterisation, which would produce discontinuities in the animation, nearby parameterisations are blended using a 3D Catmull-Rom spline interpolation [8]. This results in a single combined parameterisation, conditioned by the position of the player's sword, which is then used to compute the predicted pose. Using this method, the model can learn the details of every aspect of the animation, while maintaining a smooth motion.

For our data, we define an interaction region 120 cm wide, 200 cm tall and 70 cm deep. This space is subdivided in a 3D grid with four vertical subdivisions, four horizontal subdivisions and two depth subdivisions, totalling 32 grid points. Each grid point has a parameterisation for a base network architecture of two dense hidden layers with 55 neurons each. We use ReLU activation for the hidden layers [50].

On training, features are individually normalised, and the loss objective is defined as the sum of squared differences between the example and the output vector. The network is trained for 300 000 steps on batches of 32 examples, using a dropout rate of 0.3, a regularisation factor of 100 and Adam optimisation [34] with a learning rate of 0.0001.

INTERACTION DESIGN METHODOLOGY

Drawing on the system design described in the previous section, we summarise the general interaction design methodology provided by the Touché framework:

1. The gestures and expressions that form the interaction are defined. In our case, the gestures are simply two-handed, directional strikes, and the expressions that can be requested from the character are defending and attacking.
2. The necessary data is captured. On the one hand, gestural data is recorded, e.g. using the simple method in the previous section, although alternative strategies may be considered. Animation data is recorded with specialised motion capture equipment, according to the desired expressions.
3. The gesture recognition and animation synthesis models are trained as described, completing the physical level of the framework.
4. The semantic level is defined, designing the desired behaviour and the parameters to modulate it.
5. The realised framework is configured according to different scenarios or users.

The generality of our method allows our approach to be easily extrapolated to new situations and requirements, such as new expressions or changes in the character behaviour.

TECHNICAL EVALUATION

The first aspect of our system we evaluated is whether our data-driven models are learning successfully from the data. This is an evaluation that we can do through purely quantitative means, analysing the behaviour of the models when presented with unseen examples. The collected data is therefore split into a training set, containing 80 % of the examples, and a test set, containing the remaining 20 %. Here we present results of models trained on the first set and evaluated on the second set.

Gesture Recognition

We can assess the accuracy of the gesture recognition system by measuring the mispredictions of the model. However, we also want to have an understanding of the cases in which those mispredictions are more frequent. We therefore analysed the errors of the neural network per gesture and also across the duration of each gesture. In particular, we want to make sure that gesture classification is more reliable in the middle sections of a gesture, allowing more room for error at the beginning and end, where the boundary of the gesture may not be as precisely defined.

Figure 3 plots the prediction accuracy for gesture class and progress. The horizontal axis records the actual progression of a gesture, and the green dashed line the average accuracy that gestures are recognise at each instant during their execution. Obviously, the middle area of the gesture is easier to recognise, reaching around 80% accuracy. This is not quite a perfect detection rate, but it is sufficient for our purposes, especially considering we are using an augmented dataset where most examples have been perturbed. The light blue line represents the predicted gesture progression through time, from 0% to 100%, compared to the actual progression indicated by the dark blue line. The darker and lighter envelopes show the spread of the error, indicating respectively how far 50% and

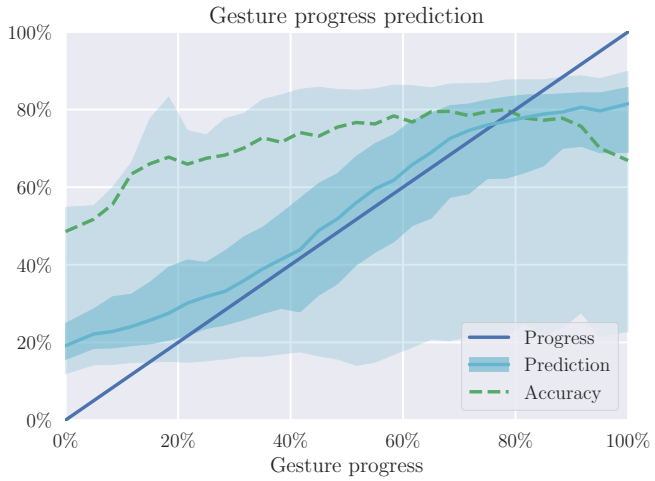

Figure 3: Gesture class and progress prediction accuracy.

95% of predictions are from the actual progress value. Again, the prediction becomes much more accurate in the middle and more uncertain towards the edges. Since here we are only interested in detecting beginnings and endings of gestures (that is, before and after 50% of their progress), this is sufficient for us.

Animation Synthesis

In order to evaluate the animation synthesis system, we need to compare a ground truth from motion capture clips with the animation produced by the model given the same user input. We expect the latter to closely resemble the former, but we need an objective measure of that. The objective loss used for training does not provide an intuitive indication of how wrong or right predictions are, so we have developed an auxiliary metric for this, which we call "pose difference". Given two poses, P^1 and P^2, we consider the positions of each of their bones, expressed as a pair of 3D points. We then have $P^1 = \{(A_1^1, B_1^1), \ldots, (A_N^1, B_N^1)\}$ and $P^2 = \{(A_1^2, B_1^2), \ldots, (A_N^2, B_N^2)\}$, N being the number of bones. The pose difference D between P^1 and P^2 is defined as:

$$D(P^1, P^2) = \sum_{i=1}^{n} \Big(\triangle(A_i^1, B_i^1, M_i) + \triangle(A_i^2, B_i^2, M_i) + \triangle(A_i^1, A_i^2, M_i) + \triangle(B_i^1, B_i^2, M_i) \Big) \quad (1)$$

where $\triangle(X, Y, Z)$ is the area of the triangle XYZ and $M_i = \frac{1}{4}(A_i^1 + B_i^1 + A_i^2 + B_i^2)$. Figure 4 visualises this formula for a single pair of bones.

The pose difference gives us an estimation of the area enclosed by each skew quadrilateral $A_i^1 B_i^1 B_i^2 A_i^2$, and therefore it is an intuitive measure (in surface units) of how different two poses are. We evaluate our defence animation model by comparing the predictions it generates with a given evaluation sequence. To do this, the model is fed with the same input it would "see" at run time (namely the position of the player's sword) and compute the pose difference between the produced pose and

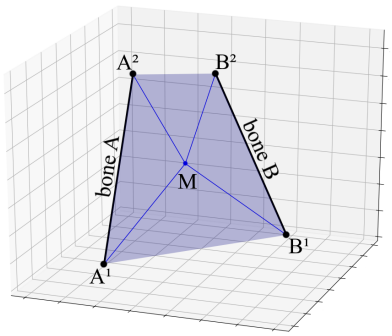

Figure 4: The pose difference measures the total distance between pairs of corresponding bones (*bone A* and *bone B*), computing the area of the four triangles from the bone ends (A^1, A^2, B^1, B^2) to the mid point (M).

the pose in the sequence. Figure 5 shows the distribution of the pose difference, measured in square meters. In total, the mean difference is 0.427 m^2, with a standard deviation of 0.267 and a median of 0.361 m^2. The main source of this error comes from the position of the sword itself (28%), as expected, followed by wrists (7% and 8%), with the rest evenly distributed across the body. Considered over the proportions of a human body and sword, this suggests that the synthesised animation does indeed follow the evaluation sequence quite closely, even though the same input, like a strike from the player, could elicit different valid reactions from the character.

USER EVALUATION

We carried out an evaluation of Touché through two user studies. Specifically, we wanted to confirm whether users find interaction in our framework superior to a typical hand-designed system, in terms of realism, interest and immersion. We also wanted to study how different configurations of our framework are perceived by players. We compared the following conditions using a counterbalanced within-participant design:

Control (C) The character is not animated by our framework but directly using motion capture clips and hand-crafted logic. In order to defend itself, the character repeatedly plays blocking animations when the player's sword approaches, chosen according to the position of the sword. This is interspersed with animations of sword strikes.

Aggressive & Unskilled (A) A character with an attack rate of 1 s^{-1} and no ability to counterattack or quick-defend.

Defensive & Skilled (D) A character with an attack rate of 0.3 s^{-1} and very good ability to counterattack and quick-defend, with a reaction time of 0.1 s.

Questionnaire Study

We first conducted a questionnaire study, where participants were presented with short videos (one minute each) of sword fighting sequences against the characters described by each of the conditions, driven by similar player interactions. The videos are from the point of view of the player and show the character walking towards the player and engaging in sword fighting, attacking and defending. After each video, participants were asked to complete a questionnaire including the

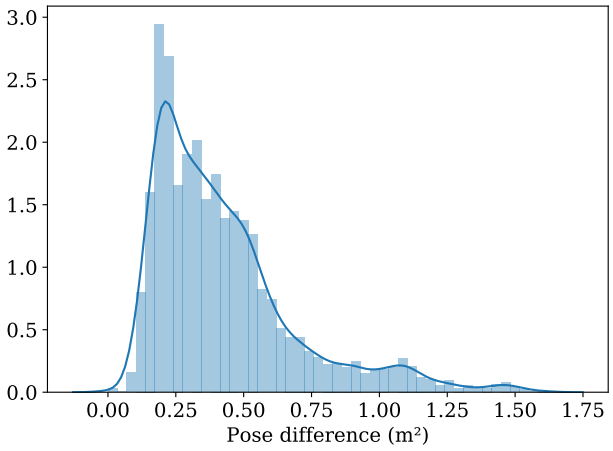

Figure 5: Normalised histogram and kernel density estimation of the distribution of pose difference between the motion predicted by our model and the evaluation data.

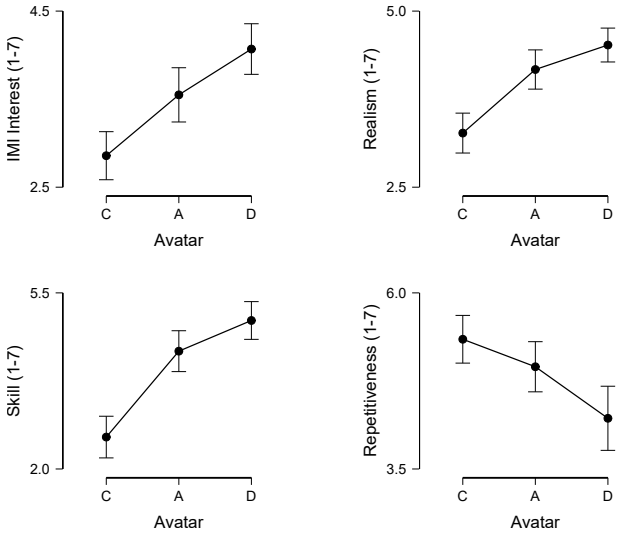

Figure 6: Interest / Enjoyment, Realism, Skill and Repetitiveness scores for the questionnaire study. The control condition C was generally inferior, while differences between A and D reflect the impact of the framework configuration.

Intrinsic Motivation Inventory (IMI) [59], Realism evaluation questions adapted from the Immersion questionnaire by Jennet et al. [29], and a set of seven-point Likert scales of adjectives to describe the character. We chose the IMI scale because it is a well-validated questionnaire about the subjective experience of a target activity, and is frequently used in gaming research, while the Immersion questionnaire is a widely-used tool to assess user experience in a virtual environment. At the end of the study participants ranked the videos by realism.

We obtained answers from 41 participants, with ages from 18 to 63 (mean 31.7, s.d. 12.4). There were 11 females, 28 males and 2 participants with other or unspecified gender. Participants were also asked to self-assess their experience in different relevant areas, on a scale from one to five. Overall, mean experience with video games was 3.59 (s.d. 1.02), with virtual reality was 2.24 (s.d. 1.09), with video games development was 2.22 (s.d. 1.39), with 3D animation was 1.78 (s.d. 0.85) and with actual sword fighting was 1.32 (s.d. 0.57).

We conducted one-way repeated-measures ANOVAs to compare the effect of the avatar on the dependent variables, using two-tailed t-tests with Holm correction to make pairwise comparisons. Figure 6 shows the results of the study. The effects of each avatar on Interest / Enjoyment ($F(80) = 17.96$, $p < 0.001$) and Realism ($F(80) = 23.67$, $p < 0.001$) were significant. The control condition C was perceived as significantly inferior in Interest / Enjoyment ($t(40) \geq 3.379$, $p \leq 0.003$) and Realism ($t(40) \geq 4.400$, $p \leq 0.001$), corroborating the value of our approach. Aligned with our expectations, we also saw significant differences for the descriptive adjectives "Skilled" ($F(80) = 37.051$, $p < 0.001$) and "Repetitive" ($F(80) = 8.862$, $p < 0.001$). Condition C was clearly considered less skilled ($t(40) \geq 5.792$, $p < 0.001$), while configuration D was viewed as the least repetitive one ($t(40) \geq 2.435$, $p \leq 0.039$). This also illustrates the effect of the design parameters on the user perception, and suggests that the animation synthesis component of the character animation, more prevalent in condition D (focused on defence), does contribute to a positive perception of the character.

In the final realism ranking question, condition C was selected as most realistic by only 1 participant (2.4%), versus condition A, selected by 14 participants (34.2%), and condition D, chosen by 24 participants (58.5%). Two participants did not provide a proper ranking.

Among the comments provided by the participants, control condition C was generally described as "unresponsive" or "clumsy", whereas condition D was described as "much more enjoyable". Various participants pointed out the lack of locomotion once the sword fight starts. This is a current limitation of our framework, which is not easy to address given that typical VR setups are limited to hand and head data capture. There were also several comments about the limitations of assessing a VR experience through a video recording, which leads us to our second study.

Interactive Study

In this study we asked volunteers to try out Touché by themselves, under the three conditions described above. Participants were asked to interact with the virtual character for about three minutes for each condition. Conditions were presented in counterbalanced order. After each interaction, participants were asked to answer the same questionnaire used for our first study (wording adapted to the interactive experience), extended with the Witmer & Singer presence questionnaire [71] (seven-point Likert scales).

We had 12 participants in this study, aged from 23 to 40 years (mean 31.9, s.d. 5.5). There were 4 female and 8 male participants. Their mean self-assessed experience with video games was 4.25 (s.d. 0.75), with virtual reality was 3.33 (s.d. 0.89),

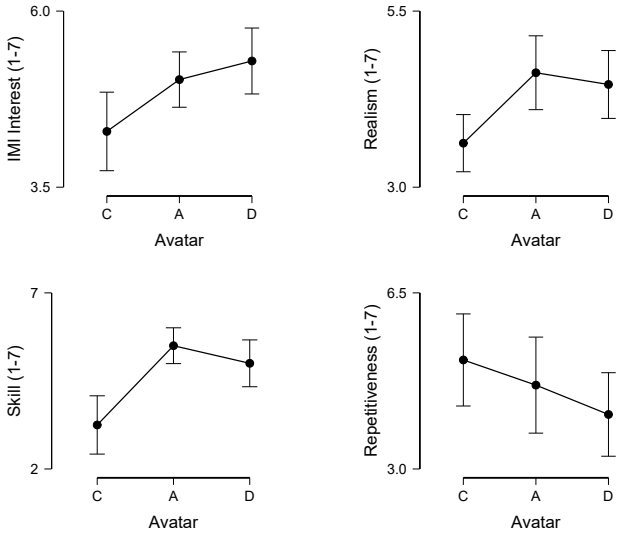

Figure 7: Interest / Enjoyment, Realism, Skill and Repetitiveness scores for the interactive study.

with video games development was 3.58 (s.d. 1.00), with 3D animation was 2.83 (s.d. 0.84) and with actual sword fighting was 1.67 (s.d. 0.78). Overall, the expertise of the participants in this study was greater in all relevant areas.

Based on the first study we hypothesised that Touché delivers more enjoyable, realistic and immersive sword fighting experiences, presenting a more skilled and less repetitive opponent behaviour. We tested these hypotheses using one-tailed t-tests, with the results shown in Figure 7 and Table 1. In general, the results reaffirm our findings from the first study. Interestingly, we find that condition C did not appear as significantly more repetitive than condition A. This, however, has an explanation in the fact that condition A (aggressive behaviour) is interleaving clips of attack animations with more frequency. This may indicate that replacing the attacking behaviour using a more adaptive approach (similar to defence animation synthesis) could improve the experience.

We found that one-way repeated-measured ANOVAs of Presence or its sub-scales did not detect a significant effect of the avatar ($p \geq 0.308$). We contend that the sense of presence is more influenced by the appearance of the environment, along with sensorial factors like sound or haptic feedback, which remained constant across all conditions and are not part of the scope of our work.

In the overall realism ranking, condition C was again the first option for only 1 participant (8.3%), while condition A was selected by 6 participants (50.0%) and condition D by 5 participants (41.7%). In spite of some differences between the results of the studies, the ranking shows a clear preference for Touché in both cases.

The participants also gave insightful comments about the different avatars. With condition C, participants felt their actions had little impact on the behaviour of the opponent, noting

Scale	Cond. 1	Cond. 2	$t(11)$	p
IMI Interest	A	C	2.353	0.019
	D	C	2.833	0.008
Realism	A	C	3.352	0.003
	D	C	3.120	0.005
Skill	A	C	5.046	< 0.001
	D	C	3.339	0.003
Repetitiveness	C	A	0.804	0.219
	C	D	1.995	0.036

Table 1: One-tailed t-tests over the results of the interactive study. All scales range from one to seven. The alternative hypotheses are that the measures for condition 1 are greater than those for condition 2.

that it "became very predictable". Condition A was on the other hand considered "very responsive", although some participants found its aggressiveness made the experience "too hard". Condition D was as well seen as "more intelligent", and its defensive attitude made it difficult to "land a hit" on the avatar. Several comments mentioned issues with the sword collision system. We used a basic collision mechanism for our studies and, although it is not part of the scope of our work, we realise this is an important aspect of the experience which might require further sophistication. The lack of hit reactions from the avatar was also brought up in multiple occasions, and reported as detrimental to overall believability. Finally, there were suggestions to improve the experience through additional elements like shields or locomotion during combat.

DISCUSSION

Our work with Touché shows that using data-driven models can be an effective approach to building interactive experiences in VR, a medium where conventional animation and design techniques cannot be easily applied. This builds on the recent trend of using machine learning and related methods in animation and other aspects of real-time interaction, which we expect to continue. By taking concepts from well-established fields, such as gesture recognition [47, 9] and animation synthesis [37, 41, 27], we have constructed an end-to-end interaction framework that is both realistic and simple to use. As computational power grows further and demand for realism in digital entertainment increases, we see this kind of aggregation of data-driven techniques as a viable path to tackling the complexity of the generation of human-like behaviour.

We have achieved our results using fairly limited data and computational resources (not even one hour of collected data in total and the power of one desktop computer). This highlights the potential of our method, as extending the framework, e.g. with different sword fighting styles or recognised gestures, would only require changing or expanding the collected data. The framework was developed in partnership with a well-known, award-winning video games studio, receiving continuous feedback from professional game developers, which informed our approach. For instance, we base our design for the semantic level on state machines, which are commonly

used for game logic and animation, because designers and animators report feeling comfortable working with them.

Our user studies corroborate that Touché is capable of producing more realistic and engaging sword fighting experiences than simple hand-made logic. The study results support the claim that data-driven models can make a significant difference in the perceived realism of an interaction, while also showing that simple, high-level control through parameters can be enough to design a varied range of experiences. We interpret this as a sign that, again, we should continue research into the automation of complex, low-level computer-interaction tasks (see [10, 26] for some good examples of this). We also recognise the undeniable importance of explicit, human-designed control of the experience, advocating simplified models that allow designers to configure only as much as they need to influence the overall interaction.

Limitations

There are some noticeable limitations to our framework, stemming from different sources. The most apparent is the restricted scope of sword fighting, which is limited to two-handed swords only and does not feature lower-body motion. Also, we did not include hit reactions in our model, which several study participants pointed out as a weakness, along with limitation in the sword collision system. More generally, the interaction based on strike gestures can also be restrictive, as opposed to considering a broader "vocabulary" including defensive poses or full-body actions from the player.

We anticipate that most of these issues can be addressed by collecting the appropriate data to emulate the desired behaviours. For example, modelling an opponent with a sword and a shield should essentially come down to recording defensive and offensive actions with these props, which our framework would then just reproduce. However, the technology itself can also be limiting in some ways. VR setups are generally restricted to a relatively small area, so more wide-ranging interactions, like a sword combat with locomotion, are in most cases just not physically possible. This is a constraint of our focus on domestic technology, but it would be interesting to extend our approach to more expansive VR installations. Issues with the collision system also fall out of our scope, but they underline the impact that disparate elements of the experience can have on each other.

Finally, we note the difficulties in carrying out evaluations of this kind of technology, especially given the lack of directly comparable prior work to use as baseline. While showing recordings of VR interactions is simple, it is also clearly insufficient by itself. However, VR sword fighting can be physically demanding for many users, so interactive studies cannot run for more than a few minutes. We found however that the combination of both kinds of studies, along with a technical evaluation, establish a reliable measure of the value of Touché.

Impact

Data-driven approaches to real-time character animation and interaction have already started to feature in major commercial productions [7, 11], and we expect this to become the norm in the near future. VR is a medium that is, in a way, starting to find its place in the market of digital entertainment, surrounded by very high expectations. There is however still a lot to explore in the intersection of these two paradigms, especially in the context of video games or game-like applications. Our results give us confidence that data-driven techniques will play an essential role in the design of realistic interactions with virtual humans in VR.

With Touché, we aim to demonstrate a practical example of this approach. The modular design we propose makes our concept easily extensible to other scenarios, and acknowledged limitations can generally be solved within it. We expect that this methodology can be straightforwardly applied not just to variants of sword fighting, such as shielded combat, but to different VR interactions such as boxing, close-range cooperation or even dancing. Game developers and animators were generally positive about the result, finding that the results were "realistic" and the framework "simplifies a lot the animation work". Given the profound complexities involved in animating virtual humans, especially in the case of VR, we see the dual physical / semantic model as an important advance that can dramatically simplify, or even make possible, the development of realistic VR experiences in the future.

CONCLUSION

We have introduced Touché, a framework for interactive sword fighting in VR. By dividing the problem into a physical and a semantic level, we have been able to design an interaction framework that minimises the amount of necessary human work while offering a reasonable design space to direct the experience. To do this, we incorporated ideas from gesture recognition and animation synthesis, using data-driven techniques to automate the most complex aspects of our system. As a result, our framework can be configured with a simple set of intuitive parameters.

We have conducted several evaluations of our framework, both through purely technical means and with user participation, showing that our framework enables an interaction that is consistently more realistic and interesting than simple hand-made logic, without requiring any additional manual animation work. Our results with Touché support the hypothesis that data-driven models will play an increasingly important role in interactive media in the future. Our framework, which focuses on the case of sword fighting, puts forward a sound methodology to tackle complex interactions in VR in general, built on concepts from different areas. We believe this integrative approach is a key aspect to its success, and one which we expect to gain more presence in the next generation of data-driven interactive systems. Our framework code, training data and demonstration videos are available online [12].[1]

ACKNOWLEDGEMENTS

This work was funded and supported by the Centre for Digital Entertainment at the University of Bath (EPSRC award EP/L016540/1) and Ninja Theory Ltd. Demonstration environment and sword model by Epic Games, Inc.

[1] https://doi.org/10.15125/BATH-00754

REFERENCES

[1] Maryam Asadi-Aghbolaghi, Albert Clapés, Marco Bellantonio, Hugo Jair Escalante, Víctor Ponce-López, Xavier Baró, Isabelle Guyon, Shohreh Kasaei, and Sergio Escalera. 2017. Deep Learning for Action and Gesture Recognition in Image Sequences: A Survey. In *Gesture Recognition*. Springer, 539–578.

[2] Jeremy N. Bailenson, Jim Blascovich, Andrew C. Beall, and Jack M. Loomis. 2003. Interpersonal Distance in Immersive Virtual Environments. *Personality and Social Psychology Bulletin* 29, 7 (2003), 819–833.

[3] Dimitrios Batras, Judith Guez, Jean-François Jégo, and Marie-Hélène Tramus. 2016. A Virtual Reality Agent-Based Platform for Improvisation Between Real and Virtual Actors Using Gestures. In *Proceedings of the 2016 Virtual Reality International Conference (VRIC '16)*. ACM, 34:1–34:4.

[4] Michael Bratman. 1987. *Intention, Plans, and Practical Reason*. Harvard University Press, Cambridge, Mass.

[5] Jan Broersen, Mehdi Dastani, Joris Hulstijn, Zisheng Huang, and Leendert van der Torre. 2001. The BOID Architecture: Conflicts Between Beliefs, Obligations, Intentions and Desires. In *Proceedings of the 4th International Conference on Autonomous Agents (AGENTS '01)*. ACM, 9–16.

[6] Armin Bruderlin and Lance Williams. 1995. Motion Signal Processing. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '95)*. ACM, 97–104.

[7] Michael Büttner. 2013. Reinforcement Learning Based Character Locomotion in Hitman: Absolution. In *Game Developers Conference 2013 (GDC 2013)*.

[8] Edwin Catmull and Raphael Rom. 1974. A Class of Local Interpolating Splines. In *Computer Aided Geometric Design*. Academic Press, 317–326.

[9] Hong Cheng, Lu Yang, and Zicheng Liu. 2016. Survey on 3D Hand Gesture Recognition. *IEEE Transactions on Circuits and Systems for Video Technology* 26, 9 (2016), 1659–1673.

[10] Loïc Ciccone, Martin Guay, Maurizio Nitti, and Robert W. Sumner. 2017. Authoring Motion Cycles. In *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation (SCA '17)*. ACM, 8:1–8:9.

[11] Simon Clavet. 2016. Motion Matching and The Road to Next-Gen Animation. In *Game Developers Conference 2016 (GDC 2016)*.

[12] Javier Dehesa and Ninja Theory Ltd. 2020. Dataset for "Touché: Data-Driven Interactive Sword Fighting in Virtual Reality". University of Bath Research Data Archive, Bath, UK. https://doi.org/10.15125/BATH-00754.

[13] Javier Dehesa, Andrew Vidler, Christof Lutteroth, and Julian Padget. 2019. Towards Data-Driven Sword Fighting Experiences in VR. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems (CHI EA '19)*. LBW2117:1–LBW2117:6.

[14] David DeVault, Ron Artstein, Grace Benn, Teresa Dey, Ed Fast, Alesia Gainer, Kallirroi Georgila, Jon Gratch, Arno Hartholt, Margaux Lhommet, Gale Lucas, Stacy Marsella, Fabrizio Morbini, Angela Nazarian, Stefan Scherer, Giota Stratou, Apar Suri, David Traum, Rachel Wood, Yuyu Xu, Albert Rizzo, and Louis-Philippe Morency. 2014. SimSensei Kiosk: A Virtual Human Interviewer for Healthcare Decision Support. In *Proceedings of the 2014 International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS '14)*. International Foundation for Autonomous Agents and Multiagent Systems, 1061–1068.

[15] Mahmoud Elmezain, Ayoub Al-Hamadi, Jörg Appenrodt, and Bernd Michaelis. 2008. A Hidden Markov Model-Based Continuous Gesture Recognition System for Hand Motion Trajectory. In *Proceedings of the 19th International Conference on Pattern Recognition (ICPR 2008)*. IEEE, 1–4.

[16] Kutluhan Erol, James A. Hendler, and Dana S. Nau. 1994. UMCP: A Sound and Complete Procedure for Hierarchical Task-Network Planning. In *Proceedings of the 2nd International Conference on Artificial Intelligence Planning Systems (AIPS '14)*, Kristian J. Hammond (Ed.). AAAI, 249–254.

[17] Sergio Escalera, Xavier Baró, Jordi Gonzàlez, Miguel A. Bautista, Meysam Madadi, Miguel Reyes, Víctor Ponce-López, Hugo J. Escalante, Jamie Shotton, and Isabelle Guyon. 2014. ChaLearn Looking at People Challenge 2014: Dataset and Results. In *Workshops at the 13th European Conference on Computer Vision (ECCV 2014) (Lecture Notes in Computer Science)*. Springer, 459–473.

[18] Sean Ryan Fanello, Ilaria Gori, Giorgio Metta, and Francesca Odone. 2017. Keep It Simple and Sparse: Real-Time Action Recognition. In *Gesture Recognition*. Springer, 303–328.

[19] Richard E. Fikes and Nils J. Nilsson. 1971. Strips: A New Approach to the Application of Theorem Proving to Problem Solving. *Artificial Intelligence* 2, 3 (1971), 189–208.

[20] Katerina Fragkiadaki, Sergey Levine, Panna Felsen, and Jitendra Malik. 2015. Recurrent Network Models for Human Dynamics. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV '15)*. 4346–4354.

[21] Michael P. Georgeff and François Felix Ingrand. 1989. Decision-Making in an Embedded Reasoning System. In *Proceedings of the 11th International Joint Conference on Artificial Intelligence*. Morgan Kaufmann, 972–978.

[22] Michael Gleicher. 2001. Motion Path Editing. In *Proceedings of the 2001 Symposium on Interactive 3D Graphics (I3D '01)*. ACM, 195–202.

[23] Rachel Heck and Michael Gleicher. 2007. Parametric Motion Graphs. In *Proceedings of the 2007 Symposium on Interactive 3D Graphics and Games (I3D '07)*. ACM, 129–136.

[24] Edmond S. L. Ho, Jacky C. P. Chan, Taku Komura, and Howard Leung. 2013. Interactive Partner Control in Close Interactions for Real-Time Applications. *ACM Trans. Multimedia Comput. Commun. Appl.* 9, 3 (July 2013), 21:1–21:19.

[25] Edmond S. L. Ho and Taku Komura. 2011. A Finite State Machine Based on Topology Coordinates for Wrestling Games. *Computer Animation and Virtual Worlds* 22, 5 (Sept. 2011), 435–443.

[26] Daniel Holden. 2018. Robust Solving of Optical Motion Capture Data by Denoising. *ACM Transactions on Graphics* 37, 4 (2018), 165:1–165:12.

[27] Daniel Holden, Taku Komura, and Jun Saito. 2017. Phase-Functioned Neural Networks for Character Control. *ACM Transactions on Graphics* 36, 4 (2017), 42:1–42:13.

[28] Lucio Ieronutti and Luca Chittaro. 2007. Employing Virtual Humans for Education and Training in X3D/VRML Worlds. *Computers & Education* 49, 1 (2007), 93–109.

[29] Charlene Jennett, Anna L. Cox, Paul Cairns, Samira Dhoparee, Andrew Epps, Tim Tijs, and Alison Walton. 2008. Measuring and Defining the Experience of Immersion in Games. *International Journal of Human-Computer Studies* 66, 9 (2008), 641–661.

[30] Sanna Kallio, Juha Kela, and Jani Mäntyjärvi. 2003. Online Gesture Recognition System for Mobile Interaction. In *Proceedings of the 2003 IEEE International Conference on Systems, Man and Cybernetics (SMC '03)*, Vol. 3. IEEE, 2070–2076.

[31] Michelle R. Kandalaft, Nyaz Didehbani, Daniel C. Krawczyk, Tandra T. Allen, and Sandra B. Chapman. 2013. Virtual Reality Social Cognition Training for Young Adults with High-Functioning Autism. *Journal of Autism and Developmental Disorders* 43, 1 (2013), 34–44.

[32] Cem Keskin, Ali Taylan Cemgil, and Lale Akarun. 2011. DTW Based Clustering to Improve Hand Gesture Recognition. In *Human Behavior Understanding*. Springer, 72–81.

[33] Sangki Kim, Gunhyuk Park, Sunghoon Yim, Seungmoon Choi, and Seungjin Choi. 2009. Gesture-Recognizing Hand-Held Interface with Vibrotactile Feedback for 3D Interaction. *IEEE Transactions on Consumer Electronics* 55, 3 (2009), 1169–1177.

[34] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR 2015)*.

[35] Jan Kolkmeier, Jered Vroon, and Dirk Heylen. 2016. Interacting with Virtual Agents in Shared Space: Single and Joint Effects of Gaze and Proxemics. In *Proceedings of the 16th International Conference on Intelligent Virtual Agents (IVA 2016)*. Springer, 1–14.

[36] Iuliia Kotseruba and John K. Tsotsos. 2016. A Review of 40 Years of Cognitive Architecture Research: Core Cognitive Abilities and Practical Applications. *arXiv:1610.08602 [cs]* (2016).

[37] Lucas Kovar and Michael Gleicher. 2004. Automated Extraction and Parameterization of Motions in Large Data Sets. *ACM Transactions on Graphics* 23, 3 (2004), 559.

[38] Lucas Kovar, Michael Gleicher, and Frédéric Pighin. 2002. Motion Graphs. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '02)*. ACM, 473–482.

[39] John E. Laird, Allen Newell, and Paul S. Rosenbloom. 1987. SOAR: An Architecture for General Intelligence. *Artificial Intelligence* 33, 1 (1987), 1–64.

[40] Jehee Lee and Kang Hoon Lee. 2006. Precomputing Avatar Behavior from Human Motion Data. *Graphical Models* 68, 2 (March 2006), 158–174.

[41] Yongjoon Lee, Kevin Wampler, Gilbert Bernstein, Jovan Popović, and Zoran Popović. 2010. Motion Fields for Interactive Character Locomotion. In *ACM SIGGRAPH Asia 2010 Papers (SIGGRAPH ASIA '10)*. ACM, 138:1–138:8.

[42] Rung-Huei Liang and Ming Ouhyoung. 1998. A Real-Time Continuous Gesture Recognition System for Sign Language. In *Proceedings of the 3rd IEEE International Conference on Automatic Face and Gesture Recognition (FG '98)*. IEEE, 558–567.

[43] Joan Llobera, Bernhard Spanlang, Giulio Ruffini, and Mel Slater. 2010. Proxemics with Multiple Dynamic Characters in an Immersive Virtual Environment. *ACM Trans. Appl. Percept.* 8, 1 (2010), 3:1–3:12.

[44] Liang Lu, Lingpeng Kong, Chris Dyer, Noah A. Smith, and Steve Renals. 2016. Segmental Recurrent Neural Networks for End-to-End Speech Recognition. *arXiv:1603.00223 [cs]* (2016).

[45] Jani Mäntyjärvi, Juha Kela, Panu Korpipää, and Sanna Kallio. 2004. Enabling Fast and Effortless Customisation in Accelerometer Based Gesture Interaction. In *Proceedings of the 3rd International Conference on Mobile and Ubiquitous Multimedia (MUM '04)*. ACM, 25–31.

[46] Aline Menin, Rafael Torchelsen, and Luciana Nedel. 2018. An Analysis of VR Technology Used in Immersive Simulations with a Serious Game Perspective. *IEEE Computer Graphics and Applications* 38, 2 (2018), 57–73.

[47] Sushmita Mitra and Tinku Acharya. 2007. Gesture Recognition: A Survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 37, 3 (2007), 311–324.

[48] Mark Mizuguchi, John Buchanan, and Tom Calvert. 2001. Data Driven Motion Transitions for Interactive Games. In *Eurographics 2001 - Short Presentations*. Eurographics Association.

[49] Pavlo Molchanov, Xiaodong Yang, Shalini Gupta, Kihwan Kim, Stephen Tyree, and Jan Kautz. 2016. Online Detection and Classification of Dynamic Hand Gestures with Recurrent 3D Convolutional Neural Networks. In *Proceedings of the 2016 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '16)*. IEEE, 4207–4215.

[50] Vinod Nair and Geoffrey E. Hinton. 2010. Rectified Linear Units Improve Restricted Boltzmann Machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning (ICML '10)*. Omnipress, 807–814.

[51] Natalia Neverova, Christian Wolf, Graham W. Taylor, and Florian Nebout. 2014. Multi-Scale Deep Learning for Gesture Detection and Localization. In *Workshops at the 13th European Conference on Computer Vision (ECCV 2014)*. Springer, 474–490.

[52] Tsukasa Noma, Liwei Zhao, and Norman I. Badler. 2000. Design of a Virtual Human Presenter. *IEEE Computer Graphics and Applications* 20, 4 (July 2000), 79–85.

[53] Felipe Pepe. 2019. *The CRPG Book: A Guide to Computer Role-Playing Games*. Bitmap Books, Bath, UK. OCLC: 1114936629.

[54] Ken Perlin. 1995. Real Time Responsive Animation with Personality. *IEEE Transactions on Visualization and Computer Graphics* 1, 1 (March 1995), 5–15.

[55] Giuseppe Raffa, Jinwon Lee, Lama Nachman, and Junehwa Song. 2010. Don't Slow Me down: Bringing Energy Efficiency to Continuous Gesture Recognition. In *Proceedings of the 2010 International Symposium on Wearable Computers (ISWC)*. IEEE, 1–8.

[56] Anand S. Rao and Michael P. Georgeff. 1991. Modeling Rational Agents within a BDI-Architecture. In *Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning (KR'91)*. Morgan Kaufmann, 473–484.

[57] Miguel Reyes, Gabriel Domínguez, and Sergio Escalera. 2011. Featureweighting in Dynamic Timewarping for Gesture Recognition in Depth Data. In *Proeedings of the 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*. IEEE, 1182–1188.

[58] Jeff Rickel and W. Lewis Johnson. 1999. Animated Agents for Procedural Training in Virtual Reality: Perception, Cognition, and Motor Control. *Applied Artificial Intelligence* 13, 4-5 (1999), 343–382.

[59] Richard M. Ryan. 1982. Control and Information in the Intrapersonal Sphere: An Extension of Cognitive Evaluation Theory. *Journal of Personality and Social Psychology* 43, 3 (1982), 450–461.

[60] Mark Sagar. 2015. BabyX. In *ACM SIGGRAPH 2015 Computer Animation Festival (SIGGRAPH '15)*. ACM, 184–184.

[61] Thomas Schlömer, Benjamin Poppinga, Niels Henze, and Susanne Boll. 2008. Gesture Recognition with a Wii Controller. In *Proceedings of the 2nd International Conference on Tangible and Embedded Interaction (TEI '08)*. ACM, 11–14.

[62] Sebastian Starke, He Zhang, Taku Komura, and Jun Saito. 2019. Neural State Machine for Character-Scene Interactions. *ACM Transactions on Graphics* 38, 6 (2019), 209:1–209:14.

[63] Nick Taubert, Martin Löffler, Nicolas Ludolph, Andrea Christensen, Dominik Endres, and Martin A. Giese. 2013. A Virtual Reality Setup for Controllable, Stylized Real-Time Interactions Between Humans and Avatars with Sparse Gaussian Process Dynamical Models. In *Proceedings of the ACM Symposium on Applied Perception (SAP '13)*. ACM, 41–44.

[64] Graham W. Taylor, Geoffrey E Hinton, and Sam T. Roweis. 2007. Modeling Human Motion Using Binary Latent Variables. In *Advances in Neural Information Processing Systems 19*. MIT Press, 1345–1352.

[65] Adrien Treuille, Yongjoon Lee, and Zoran Popović. 2007. Near-Optimal Character Animation with Continuous Control. *ACM Transactions on Graphics* 36, 3 (2007), 7:1–7:7.

[66] Eleni Tsironi, Pablo Barros, and Stefan Wermter. 2016. Gesture Recognition with a Convolutional Long Short-Term Memory Recurrent Neural Network. In *Proceedings of the 24th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN 2016)*. Ciaco - i6doc.com, 213–218.

[67] David Vogt, Steve Grehl, Erik Berger, Heni Ben Amor, and Bernhard Jung. 2014. A Data-Driven Method for Real-Time Character Animation in Human-Agent Interaction. In *Proceedings of the 14th International Conference on Intelligent Virtual Agents (IVA 2014)*. Springer, 463–476.

[68] Sy Bor Wang, Ariadna Quattoni, Louis-Philippe Morency, David Demirdjian, and Trevor Darrell. 2006. Hidden Conditional Random Fields for Gesture Recognition. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '06)*, Vol. 2. IEEE, 1521–1527.

[69] John Weissmann and Ralf Salomon. 1999. Gesture Recognition for Virtual Reality Applications Using Data Gloves and Neural Networks. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN '99)*, Vol. 3. IEEE, 2043–2046.

[70] Andrew Witkin and Zoran Popovic. 1995. Motion Warping. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '95)*. ACM, 105–108.

[71] Bob G. Witmer and Michael J. Singer. 1998. Measuring Presence in Virtual Environments: A Presence Questionnaire. *Presence: Teleoperators and Virtual Environments* 7, 3 (1998), 225–240.

[72] Sebastien C. Wong, Adam Gatt, Victor Stamatescu, and Mark D. McDonnell. 2016. Understanding Data Augmentation for Classification: When to Warp?. In *Proceedings of the 2016 International Conference on Digital Image Computing: Techniques and Applications (DICTA '16)*. IEEE, 1–6.

[73] Deyou Xu. 2006. A Neural Network Approach for Hand Gesture Recognition in Virtual Reality Driving Training System of SPG. In *Proceedings of the 18th International Conference on Pattern Recognition (ICPR '06)*, Vol. 3. IEEE, 519–522.

[74] Junji Yamato, Jun Ohya, and Kenichiro Ishii. 1992. Recognizing Human Action in Time-Sequential Images Using Hidden Markov Model. In *Proceedings of the 1992 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '92)*. IEEE, 379–385.

[75] Ying Yin and Randall Davis. 2014. Real-Time Continuous Gesture Recognition for Natural Human-Computer Interaction. In *Proceedings of the 2014 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC '14)*. 113–120.

[76] Fisher Yu and Vladlen Koltun. 2015. Multi-Scale Context Aggregation by Dilated Convolutions. In *Proceedings of the 4rd International Conference on Learning Representations (ICLR 2016)*.

[77] He Zhang, Sebastian Starke, Taku Komura, and Jun Saito. 2018. Mode-Adaptive Neural Networks for Quadruped Motion Control. *ACM Transactions on Graphics* 37, 4 (2018), 145:1–145:11.